# Smart Device Trends for Bellabeat

F. Dabrowski

2024-04-13

## Objective

The project's objective is to analyze smart device usage data to uncover trends, patterns, and provide business recommendations for product development and marketing strategy.

## Data sources

Data for analysis, sourced from the Bellabeat analytics team, spans from March 12 to May 12, 2016. The dataset, derived directly from FitBit trackers, consists of 35 responders' minute-level usage, ensuring reliability with a Kaggle score of 8.75/10. It's cited in various research and is CC0-licensed, allowing unrestricted use, with data integrity verified through metadata checks and pre-cleaning.

Concerns include the small sample size of 35 respondents, data from 2016, and potential skewness, mitigated by rigorous data verification and cleaning processes.

## Tools

Data analysis was conducted using the following tools:

- R: Utilized for date cleaning, analysis and visualization.
- SQL used once in order to merge daily_activity and daily_sleep datasets.

Data integrity was ensured through the following measures:

- Data pre-processed by the provider.
- Removal of duplicates.
- Verification of dates and outliers.
- Retained rows with missing values to indicate inactive tracker use.
- Ensured consistent date format and column names.

Methods for data verification included:

- Running integrity checks.
- Reviewing summary statistics.
- Conducting visual inspection of the data.

## Packages

- **tidyverse** - a collection of R packages designed for data science that includes data manipulation, plotting, and more.
- **here** - simplifies the specification of file paths that work across different operating systems.
- **skimr** - provides compact and flexible summaries of data frames.
- **janitor** - tools for cleaning and examining data frames in a simple way.
- **lubridate** - makes it easier to work with dates and times in R.
- **knitr** - allows for dynamic report generation with R, integrating R code into documents.

- **dplyr** - a grammar of data manipulation, providing a consistent set of verbs that help you solve the most common data manipulation challenges.
- **scales** - provides methods for automatically and manually transforming data for visualization purposes.
- **corrplot** - visualizes correlation matrices, typically through a lower triangle plot with colored cells representing different levels of correlation.

## Dataset import: loading and importing data

Datasets uploaded manually, downloaded from Kaggle:

```
daily_activity <- read_csv(here("bellabeat", "dailyActivity_merged.csv"))
daily_calories <- read_csv(here("bellabeat", "dailyCalories_merged.csv"))
daily_intensities <- read_csv(here("bellabeat", "dailyIntensities_merged.csv"))
daily_sleep <- read_csv(here("bellabeat", "sleepDay_merged.csv"))
daily_steps <- read_csv(here("bellabeat", "dailySteps_merged.csv"))
hourly_calories <- read_csv(here("bellabeat", "hourlyCalories_merged.csv"))
hourly_intensities <- read_csv(here("bellabeat", "hourlyIntensities_merged.csv"))
hourly_steps <- read_csv(here("bellabeat", "hourlySteps_merged.csv"))
weight_log <- read_csv(here("bellabeat", "weightLogInfo_merged.csv"))
```

## Data cleaning

### Participant count verification

```
n_unique(daily_activity$Id)
```

```
## [1] 33
```

```
n_unique(daily_sleep$Id)
```

```
## [1] 24
```

```
n_unique(daily_steps$Id)
```

```
## [1] 33
```

```
n_unique(hourly_calories$Id)
```

```
## [1] 33
```

```
n_unique(hourly_intensities$Id)
```

```
## [1] 33
```

```
n_unique(hourly_steps$Id)
```

```
## [1] 33
```

```
n_unique(weight_log$Id)
```

```
## [1] 8
```

Removing the weight_log dataset as the sample size of 8 is too small to provide reliable insights during analysis.

```
rm(weight_log)
```

### Data integrity check: identifying duplicates and missing values

```
daily_activity <- daily_activity %>%
  distinct() %>%
  drop_na()
daily_calories <- daily_calories %>%
  distinct() %>%
  drop_na()
daily_intensities <- daily_intensities %>%
  distinct() %>%
  drop_na()
daily_sleep <- daily_sleep %>%
  distinct() %>%
  drop_na()
daily_steps <- daily_steps %>%
  distinct() %>%
  drop_na()
hourly_calories <- hourly_calories %>%
  distinct() %>%
  drop_na()
hourly_intensities <- hourly_intensities %>%
  distinct() %>%
  drop_na()
hourly_steps <- hourly_steps %>%
  distinct() %>%
  drop_na()
```

### Standardizing column names

```
clean_names(daily_activity)
```

```
## # A tibble: 940 x 15
##            id activity_date total_steps total_distance tracker_distance
##         <dbl> <chr>               <dbl>          <dbl>            <dbl>
##  1 1503960366 4/12/2016           13162           8.5              8.5
##  2 1503960366 4/13/2016           10735           6.97             6.97
##  3 1503960366 4/14/2016           10460           6.74             6.74
##  4 1503960366 4/15/2016            9762           6.28             6.28
##  5 1503960366 4/16/2016           12669           8.16             8.16
##  6 1503960366 4/17/2016            9705           6.48             6.48
##  7 1503960366 4/18/2016           13019           8.59             8.59
##  8 1503960366 4/19/2016           15506           9.88             9.88
##  9 1503960366 4/20/2016           10544           6.68             6.68
## 10 1503960366 4/21/2016            9819           6.34             6.34
## # i 930 more rows
## # i 10 more variables: logged_activities_distance <dbl>,
## #   very_active_distance <dbl>, moderately_active_distance <dbl>,
## #   light_active_distance <dbl>, sedentary_active_distance <dbl>,
## #   very_active_minutes <dbl>, fairly_active_minutes <dbl>,
## #   lightly_active_minutes <dbl>, sedentary_minutes <dbl>, calories <dbl>
```

```
daily_activity <- rename_with(daily_activity, tolower)
clean_names(daily_calories)
```

```
## # A tibble: 940 x 3
##            id activity_day calories
##         <dbl> <chr>           <dbl>
##  1 1503960366 4/12/2016        1985
##  2 1503960366 4/13/2016        1797
##  3 1503960366 4/14/2016        1776
##  4 1503960366 4/15/2016        1745
##  5 1503960366 4/16/2016        1863
##  6 1503960366 4/17/2016        1728
##  7 1503960366 4/18/2016        1921
##  8 1503960366 4/19/2016        2035
##  9 1503960366 4/20/2016        1786
## 10 1503960366 4/21/2016        1775
## # i 930 more rows
```

```r
daily_calories <- rename_with(daily_calories, tolower)
clean_names(daily_intensities)
```

```
## # A tibble: 940 x 10
##            id activity_day sedentary_minutes lightly_active_minutes
##         <dbl> <chr>                    <dbl>                  <dbl>
##  1 1503960366 4/12/2016                  728                    328
##  2 1503960366 4/13/2016                  776                    217
##  3 1503960366 4/14/2016                 1218                    181
##  4 1503960366 4/15/2016                  726                    209
##  5 1503960366 4/16/2016                  773                    221
##  6 1503960366 4/17/2016                  539                    164
##  7 1503960366 4/18/2016                 1149                    233
##  8 1503960366 4/19/2016                  775                    264
##  9 1503960366 4/20/2016                  818                    205
## 10 1503960366 4/21/2016                  838                    211
## # i 930 more rows
## # i 6 more variables: fairly_active_minutes <dbl>, very_active_minutes <dbl>,
## #   sedentary_active_distance <dbl>, light_active_distance <dbl>,
## #   moderately_active_distance <dbl>, very_active_distance <dbl>
```

```r
daily_intensities <- rename_with(daily_intensities, tolower)
clean_names(daily_sleep)
```

```
## # A tibble: 410 x 5
##          id sleep_day total_sleep_records total_minutes_asleep total_time_in_bed
##       <dbl> <chr>                   <dbl>                <dbl>             <dbl>
##  1   1.50e9 4/12/201~                   1                  327               346
##  2   1.50e9 4/13/201~                   2                  384               407
##  3   1.50e9 4/15/201~                   1                  412               442
##  4   1.50e9 4/16/201~                   2                  340               367
##  5   1.50e9 4/17/201~                   1                  700               712
##  6   1.50e9 4/19/201~                   1                  304               320
##  7   1.50e9 4/20/201~                   1                  360               377
##  8   1.50e9 4/21/201~                   1                  325               364
##  9   1.50e9 4/23/201~                   1                  361               384
## 10   1.50e9 4/24/201~                   1                  430               449
## # i 400 more rows
```

```r
daily_sleep <- rename_with(daily_sleep, tolower)
clean_names(daily_steps)
```

```
## # A tibble: 940 x 3
##            id activity_day step_total
##         <dbl> <chr>             <dbl>
##  1 1503960366 4/12/2016         13162
##  2 1503960366 4/13/2016         10735
##  3 1503960366 4/14/2016         10460
##  4 1503960366 4/15/2016          9762
##  5 1503960366 4/16/2016         12669
##  6 1503960366 4/17/2016          9705
##  7 1503960366 4/18/2016         13019
##  8 1503960366 4/19/2016         15506
##  9 1503960366 4/20/2016         10544
## 10 1503960366 4/21/2016          9819
## # i 930 more rows
```

```
daily_steps <- rename_with(daily_steps, tolower)
clean_names(hourly_calories)
```

```
## # A tibble: 22,099 x 3
##            id activity_hour         calories
##         <dbl> <chr>                    <dbl>
##  1 1503960366 4/12/2016 12:00:00 AM       81
##  2 1503960366 4/12/2016 1:00:00 AM        61
##  3 1503960366 4/12/2016 2:00:00 AM        59
##  4 1503960366 4/12/2016 3:00:00 AM        47
##  5 1503960366 4/12/2016 4:00:00 AM        48
##  6 1503960366 4/12/2016 5:00:00 AM        48
##  7 1503960366 4/12/2016 6:00:00 AM        48
##  8 1503960366 4/12/2016 7:00:00 AM        47
##  9 1503960366 4/12/2016 8:00:00 AM        68
## 10 1503960366 4/12/2016 9:00:00 AM       141
## # i 22,089 more rows
```

```
hourly_calories <- rename_with(hourly_calories, tolower)
clean_names(hourly_intensities)
```

```
## # A tibble: 22,099 x 4
##            id activity_hour         total_intensity average_intensity
##         <dbl> <chr>                           <dbl>             <dbl>
##  1 1503960366 4/12/2016 12:00:00 AM              20             0.333
##  2 1503960366 4/12/2016 1:00:00 AM                8             0.133
##  3 1503960366 4/12/2016 2:00:00 AM                7             0.117
##  4 1503960366 4/12/2016 3:00:00 AM                0             0
##  5 1503960366 4/12/2016 4:00:00 AM                0             0
##  6 1503960366 4/12/2016 5:00:00 AM                0             0
##  7 1503960366 4/12/2016 6:00:00 AM                0             0
##  8 1503960366 4/12/2016 7:00:00 AM                0             0
##  9 1503960366 4/12/2016 8:00:00 AM               13             0.217
## 10 1503960366 4/12/2016 9:00:00 AM               30             0.5
## # i 22,089 more rows
```

```
hourly_intensities <- rename_with(hourly_intensities, tolower)
clean_names(hourly_steps)
```

```
## # A tibble: 22,099 x 3
##            id activity_hour         step_total
```

```
##         <dbl> <chr>                       <dbl>
##  1 1503960366 4/12/2016 12:00:00 AM         373
##  2 1503960366 4/12/2016 1:00:00 AM          160
##  3 1503960366 4/12/2016 2:00:00 AM          151
##  4 1503960366 4/12/2016 3:00:00 AM            0
##  5 1503960366 4/12/2016 4:00:00 AM            0
##  6 1503960366 4/12/2016 5:00:00 AM            0
##  7 1503960366 4/12/2016 6:00:00 AM            0
##  8 1503960366 4/12/2016 7:00:00 AM            0
##  9 1503960366 4/12/2016 8:00:00 AM          250
## 10 1503960366 4/12/2016 9:00:00 AM         1864
## # i 22,089 more rows
```

```
hourly_steps <- rename_with(hourly_steps, tolower)
```

### Standardizing dates and time formats

```
daily_activity <- daily_activity %>%
  rename(date = activitydate) %>%
  mutate(date = as_date(date, format = "%m/%d/%Y"))
daily_calories <- daily_calories %>%
  rename(date = activityday) %>%
  mutate(date = as_date(date, format = "%m/%d/%Y"))
 daily_intensities <- daily_intensities %>%
   rename(date = activityday) %>%
   mutate(date = as_date(date, format = "%m/%d/%Y"))
 daily_sleep <- daily_sleep %>%
   rename(date = sleepday) %>%
   mutate(date = as_date(date, format = "%m/%d/%Y %I:%M:%S %p", tz = Sys.timezone()))
 daily_steps <- daily_steps %>%
   rename(date = activityday) %>%
   mutate(date = as_date(date, format = "%m/%d/%Y", tz = Sys.timezone()))
 hourly_calories <- hourly_calories %>%
  rename(date_time = activityhour) %>%
  mutate(date_time = as.POSIXct(date_time, format ="%m/%d/%Y %I:%M:%S %p" , tz=Sys.timezone()))
 hourly_intensities <- hourly_intensities %>%
  rename(date_time = activityhour) %>%
  mutate(date_time = as.POSIXct(date_time, format ="%m/%d/%Y %I:%M:%S %p" , tz=Sys.timezone()))
 hourly_steps<- hourly_steps %>%
  rename(date_time = activityhour) %>%
  mutate(date_time = as.POSIXct(date_time, format ="%m/%d/%Y %I:%M:%S %p" , tz=Sys.timezone()))
```

### Integration: merging active and sleep datasets

There is an error while merging daily_activity and daily_sleep datasets. I'm going to export the data frames and join them in SQL.

```
daily_activity_sleep <- read_csv(here("bellabeat","daily_activity_sleep.csv"))
```

```
## Rows: 410 Columns: 19
## -- Column specification --------------------------------------------------------
## Delimiter: ","
## chr   (1): key
## dbl  (17): id, totalsteps, totaldistance, trackerdistance, loggedactivitiesd...
## date  (1): date
```

```
##
## i Use `spec()` to retrieve the full column specification for this data.
## i Specify the column types or set `show_col_types = FALSE` to quiet this message.
```

A check for duplicates and blanks in new file + naming

```
daily_activity_sleep <- daily_activity_sleep %>%
  distinct() %>%
  drop_na()
clean_names(daily_activity_sleep)
```

```
## # A tibble: 410 x 19
##              id date       totalsteps totaldistance trackerdistance
##           <dbl> <date>          <dbl>         <dbl>           <dbl>
##  1 8053475328 2016-04-20      15108         12.2            12.2
##  2 8053475328 2016-04-23      22359         17.2            17.2
##  3 8053475328 2016-05-07      19769         15.7            15.7
##  4 1644430081 2016-05-08       6724          4.89            4.89
##  5 1644430081 2016-04-29       3176          2.31            2.31
##  6 1644430081 2016-05-02       3758          2.73            2.73
##  7 1644430081 2016-04-30      18213         13.2            13.2
##  8 4558609924 2016-05-01       3428          2.27            2.27
##  9 4558609924 2016-04-21      13743          9.08            9.08
## 10 4558609924 2016-04-29       7833          5.18            5.18
## # i 400 more rows
## # i 14 more variables: loggedactivitiesdistance <dbl>,
## #   veryactivedistance <dbl>, moderatelyactivedistance <dbl>,
## #   lightactivedistance <dbl>, sedentaryactivedistance <dbl>,
## #   veryactiveminutes <dbl>, fairlyactiveminutes <dbl>,
## #   lightlyactiveminutes <dbl>, sedentaryminutes <dbl>, calories <dbl>,
## #   key <chr>, total_sleep_records <dbl>, total_minutes_asleep <dbl>, ...
```

```
daily_activity_sleep <- rename_with(daily_activity_sleep, tolower)
```

#### Adding weekdays to daily_activity

```
daily_activity$weekday <- weekdays(daily_activity$date)
```

Also, creating a factor to put them in order

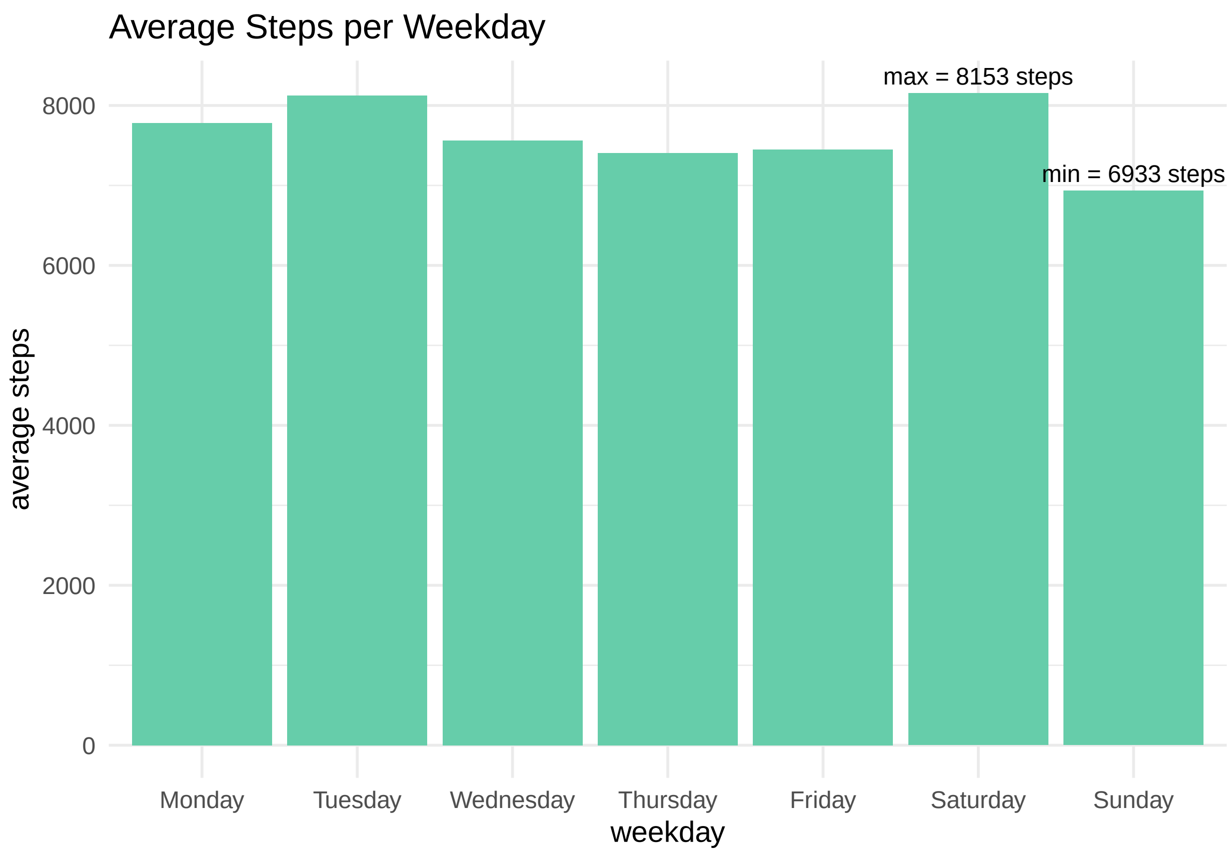
```
daily_activity$weekday <- factor(daily_activity$weekday,
                                  levels = c("Monday", "Tuesday", "Wednesday",
                                             "Thursday", "Friday", "Saturday", "Sunday"), ordered = '
```

### Activity analysis

#### Average number of steps per day

```
daily_activity %>%
    group_by(weekday) %>%
    summarise(avg_steps = mean(totalsteps, na.rm = TRUE)) %>%
    ggplot(aes(x = weekday, y = avg_steps)) +
    geom_col(fill = "aquamarine3") +
  geom_text(aes(label = ifelse(avg_steps == max(avg_steps),
                               paste("max =", round(avg_steps), "steps"),
                               ifelse(avg_steps == min(avg_steps),
```

```r
                                   paste("min =", round(avg_steps), "steps"), ""))),
            vjust = -0.5, color = "black", size = 3) +
    labs(title = "Average Steps per Weekday",
         x = "weekday",
         y = "average steps") +
    theme_minimal()
```




The chart displays the average number of steps taken on each day of the week. Saturday shows the highest average steps and Sunday the lowest.

Recommendations:

- Introduce a weekend challenge to encourage more steps on Sundays.
- Promote a daily steps goal of 10,000 steps, emphasizing increased activity on weekends.

#### Relation of steps to calories consumpion

```r
daily_activity %>%
    filter(totalsteps > 100 & totalsteps < 25000) %>%
    ggplot(aes(x = totalsteps, y = calories)) +
    geom_point(color = "black", size = 0.9) +
    geom_smooth() +
    labs(
        title = "Number of Steps vs. Calories Burned",
        x = "number of steps",
        y = "calories burned"
    ) +
    theme_minimal()
```

```
## `geom_smooth()` using method = 'loess' and formula = 'y ~ x'
```


There is a moderately strong positive correlation (approximately 0.6) between the number of steps taken and the calories burned.

Recommendations:

- Develop features in the app that provide real-time feedback on calories burned based on steps taken to motivate users.
- Use this data to personalize user goals for more effective health outcomes.

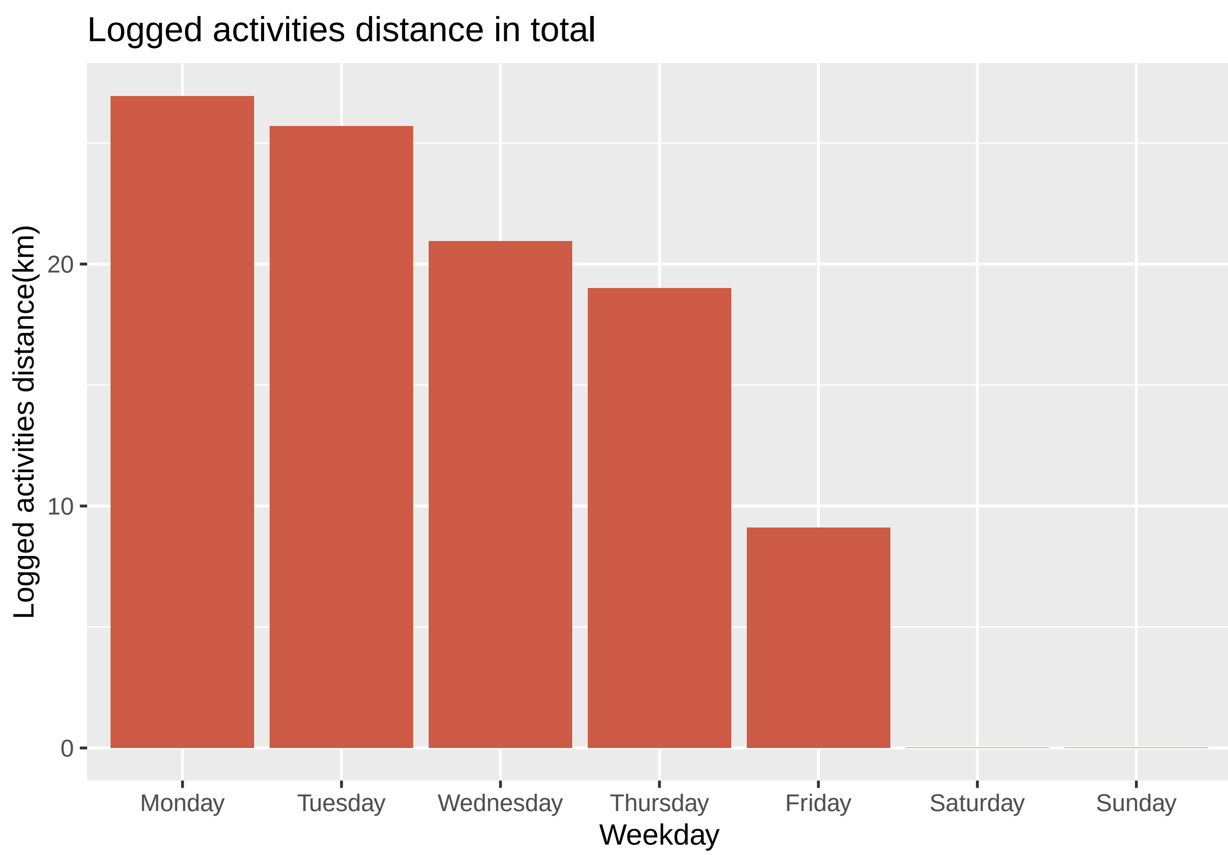
Note: outliers of daily step counts below 100 and above 25,000 have been filtered out for improved readability of the chart. However, they were included in the correlation calculations to provide a comprehensive analysis.

```
round(cor(daily_activity$totalsteps, daily_activity$calories, use = "complete.obs"),1)
```

```
## [1] 0.6
```

**Logged activities check**

```
ggplot(daily_activity, aes(x = weekday, y = loggedactivitiesdistance)) +
           geom_col(fill = "coral3") +
  labs(
    title = "Logged activities distance in total",
    x = "Weekday",
    y = "Logged activities distance(km)"
  )
```




The data reveals an absence of logged activities on Saturdays and Sundays. This could indicate a potential error in the data collection process.

Recommendations:

- Investigate and resolve the issue of missing activity logs on weekends.
- Encourage more consistent logging of activities through user reminders or incentives.

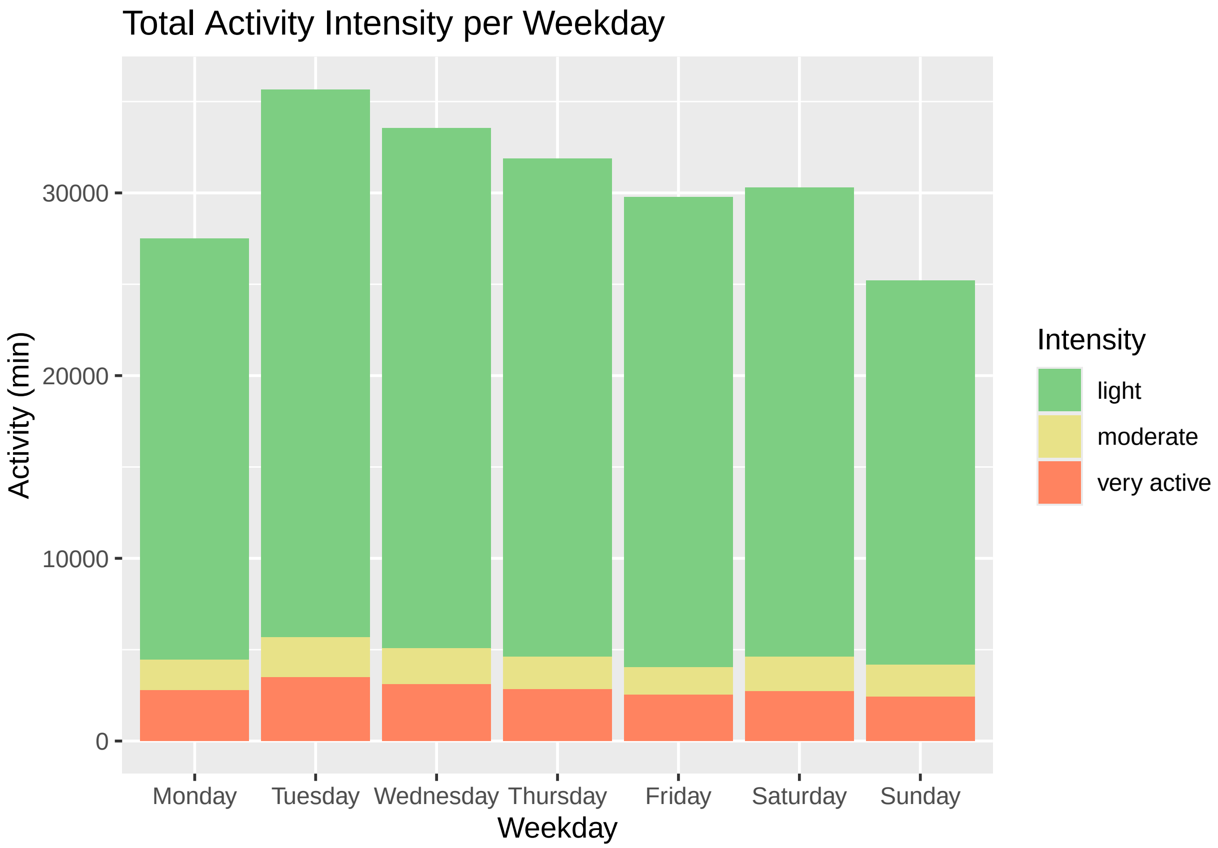
### Activity levels

Let's start with creating a vector which will set intensities ordinarily:

```
activity_levels <- c("lightlyactiveminutes", "fairlyactiveminutes", "veryactiveminutes")
```

```
daily_activity %>%
  select(date, weekday, lightlyactiveminutes, fairlyactiveminutes, veryactiveminutes) %>%
  pivot_longer(
    cols = c(lightlyactiveminutes, fairlyactiveminutes, veryactiveminutes),
    names_to = "activitytype",
    values_to = "minutes"
  ) %>%
  ggplot(aes(x = weekday, y = minutes, fill = factor(activitytype, levels = activity_levels))) +
  geom_col() +
  labs(
    title = "Total Activity Intensity per Weekday",
    x = "Weekday",
    y = "Activity (min)",
    fill = "Intensity"
  ) +
```

```
  scale_fill_manual(
    values = c("#7DCE82", "#E8E288", "#FF8360"),
    labels = c("light", "moderate", "very active")
  )
```




The data suggests that Tuesdays see the highest total minutes of activity, indicating a peak in overall activity levels.

Recommendations:

- Tailor promotional activities to increase moderate and very active minutes, especially on days with lower activity levels.
- Launch campaigns or challenges at the beginning of the week to boost overall weekly activity.

```
daily_activity_summary <- daily_activity %>%
    select(lightlyactiveminutes, fairlyactiveminutes, veryactiveminutes) %>%
    pivot_longer(
        cols = c(lightlyactiveminutes, fairlyactiveminutes, veryactiveminutes),
        names_to = "activitytype",
        values_to = "minutes"
    ) %>%
    group_by(activitytype) %>%
    summarise(across(everything(), ~ round(mean(.), 1)))
```

#### Activity piechart

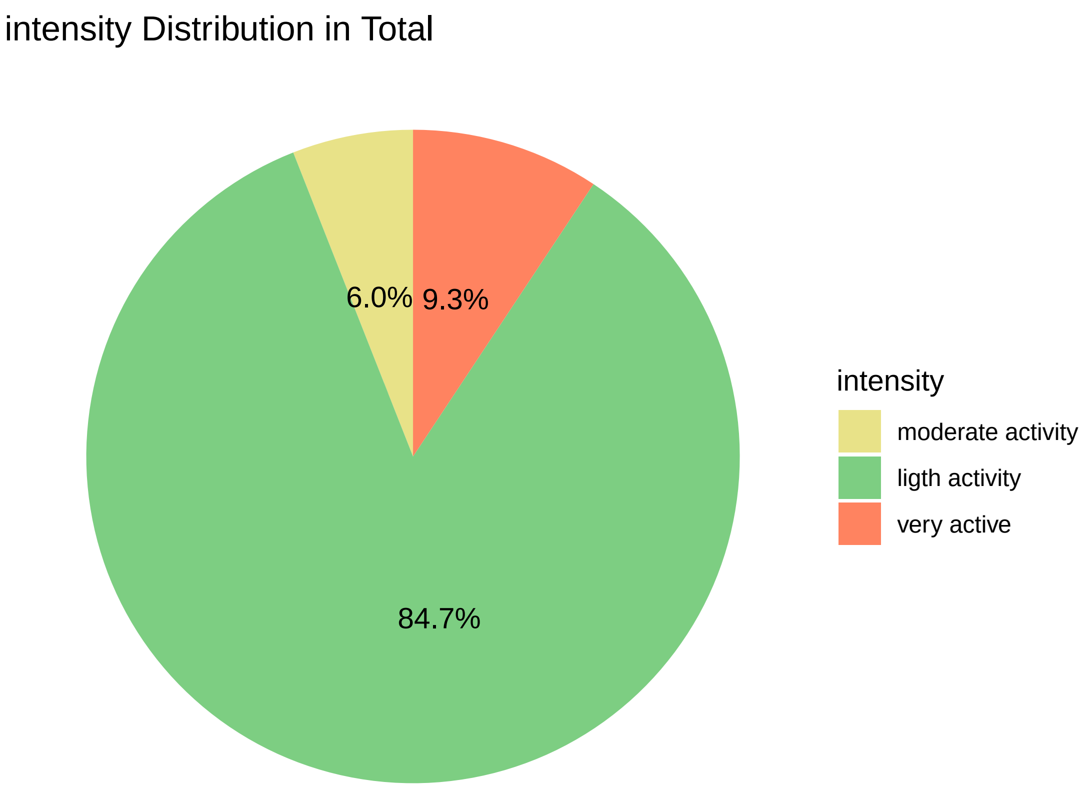
```
# Custom color values and labels
color_values <- c("lightlyactiveminutes" = "#7DCE82", "fairlyactiveminutes" = "#E8E288", "veryactivemin
```

```r
label_values <- c("moderate activity", "ligth activity", "very active")

# adding total to the dataset
daily_activity_summary$total = sum(daily_activity_summary$minutes)

# Recalculate the percentage just in case and format it
daily_activity_summary_transformed <- daily_activity_summary %>%
  mutate(percentage = minutes / total) %>%
  mutate(labels = percent(percentage))

# Plotting a pie chart
ggplot(daily_activity_summary_transformed, aes(x = "", y = percentage, fill = activitytype)) +
  geom_bar(stat = "identity", width = 1) +
  coord_polar(theta = "y") +
  theme_void() +
  scale_fill_manual(values = color_values, labels = label_values) +
  labs(
    title = "intensity Distribution in Total",
    fill = "intensity") +
  geom_text(aes(label = scales::percent(percentage)), position = position_stack(vjust = 0.45), color =
```




The chart shows a dominant 84.7% of activity classified as lightly active, with very active and fairly active minutes accounting for just 9.3% and 6% respectively. This distribution highlights a strong preference for lower-intensity activities among participants.

Recommendations:

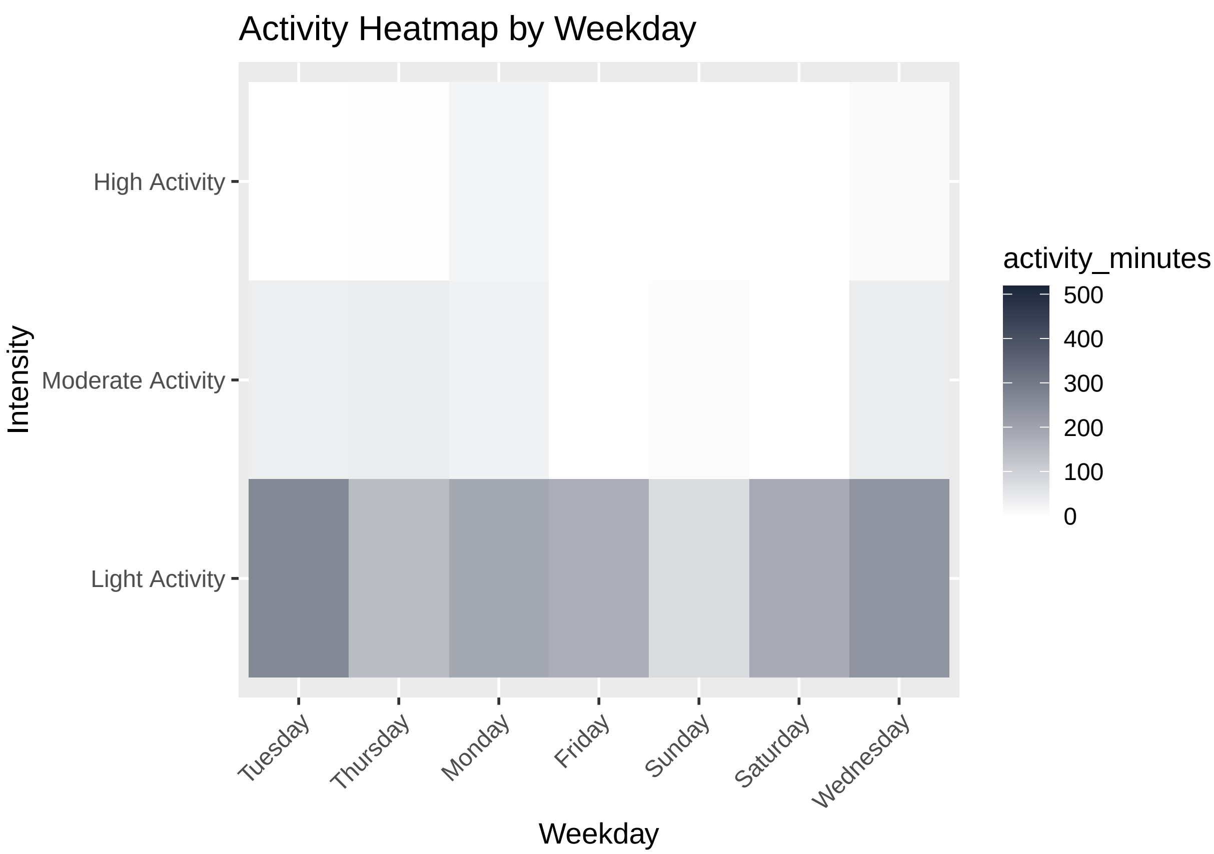
- Enhance Light Activities: To keep users engaged, consider enhancing lightly active options with new, innovative activities that add variety while staying accessible.
- Promote Higher Intensity Engagement: Increase the visibility and appeal of higher-intensity activities

through targeted promotions and incentives, aiming to capture interest from users seeking more challenging exercises.

### Activity heatmap

```
daily_activity_sleep$weekday <- weekdays(daily_activity_sleep$date)

daily_activity_sleep %>%
    pivot_longer(
        cols = c(lightlyactiveminutes, fairlyactiveminutes, veryactiveminutes),
        names_to = "activity_type",
        values_to = "activity_minutes"
    ) %>%
    mutate(activity_type = factor(activity_type,
                                  levels = c("lightlyactiveminutes", "fairlyactiveminutes", "veryactive
                                  labels = c("Light Activity", "Moderate Activity", "High Activity")))
    ggplot(aes(x = fct_rev(fct_inorder(weekday)), y = activity_type, fill = activity_minutes)) +
    geom_tile() +
    scale_fill_gradientn(colors = c("white", "#1B263B"), values = scales::rescale(c(0, 10))) +
    labs(title = "Activity Heatmap by Weekday",
         x = "Weekday",
         y = "Intensity") +
    theme(axis.text.x = element_text(angle = 45, hjust = 1))
```




Alternative version of a previous chart.

Recommendations:

- Use the heatmap insights to plan targeted activity boosts, like midweek energizers.
- Encourage balanced activity throughout the week with varied intensity levels to keep users engaged.

#### Weekday vs. weekend comparision

```
daily_activity$totalactivitymins <- daily_activity$veryactiveminutes + daily_activity$fairlyactiveminute

daily_activity %>%
  mutate(day_type = ifelse(weekday %in% c("Saturday", "Sunday"), "Weekend", "Weekday")) %>%
  ggplot(aes(x = day_type, y = totalactivitymins, fill = day_type)) +
  geom_boxplot() +
  labs(title = "Weekday vs. Weekend Activity Comparison", x = "Day Type", y = "Total Activity Minutes",
  theme_minimal()
```

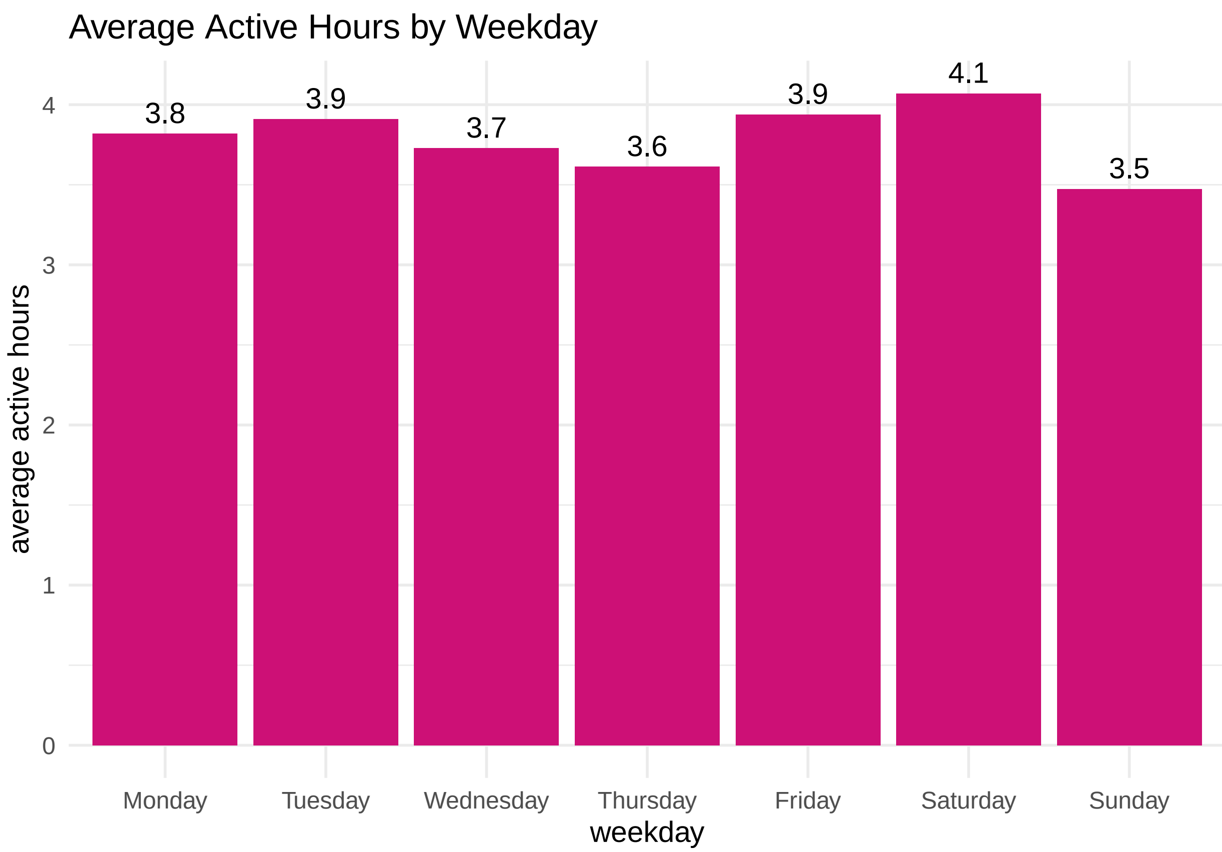

This boxplot illustrates the distribution of total activity minutes during weekdays versus weekends. The medians are similar, but the variability is greater on weekends, indicating that weekend activity levels are less consistent among individuals.

Recommendations:

- Implement weekend motivational programs: To stabilize the variability in weekend activity, introducing motivational challenges or targeted promotions could engage users to maintain or even increase their activity levels.
- Promote consistent daily targets: Encouraging users to set and meet daily activity targets could help maintain steady activity levels throughout the week, reducing the drop-off seen on weekends.

#### Average active time by weekday

```r
daily_activity %>%
    na.omit() %>%
    group_by(weekday) %>%
    summarise(avg_active_hours = mean(totalactivitymins) / 60) %>%
    ggplot(aes(x = weekday, y = avg_active_hours, fill = "deeppink3")) +
    geom_col() +
    geom_text(aes(label = sprintf("%.1f", avg_active_hours)), vjust = -0.5) +
    scale_fill_manual(values = "deeppink3", guide = "none") +
    labs(title = "Average Active Hours by Weekday",
         x = "weekday",
         y = "average active hours") +
    theme_minimal()
```



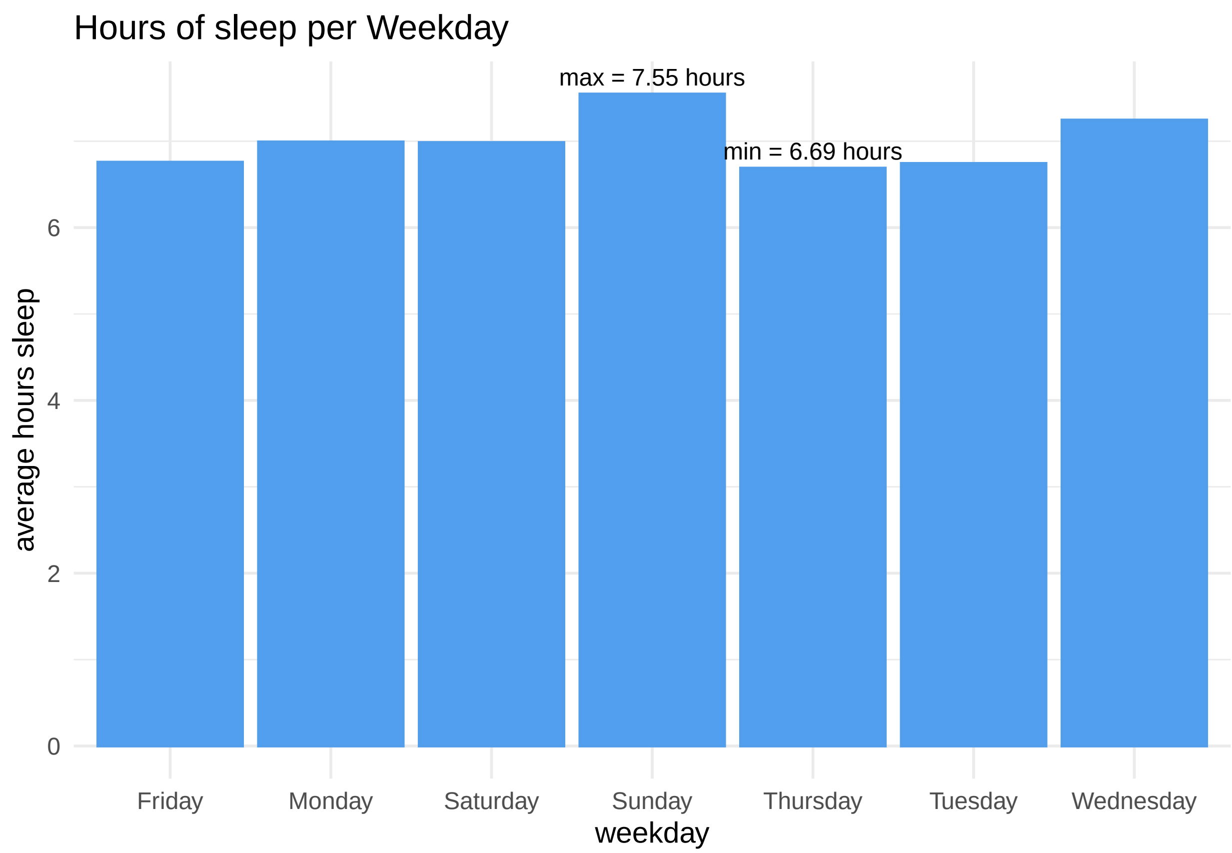

The bar chart presents a slight fluctuation in daily active hours over the week, with Wednesday marking the lowest average at 3.6 hours and Saturday the highest at 4.1 hours. The beginning and end of the workweek show marginally reduced activity. However, globally there are no significant fluctuation in daily active hours.

## Sleep

This series of charts, including bar charts and distribution graphs, explores patterns in sleep duration and quality across the week.

#### Hours of sleep per weekday

```r
daily_activity_sleep %>%
    group_by(weekday) %>%
    summarise(avg_sleep = mean(totalminutesasleep, na.rm = TRUE)/60) %>%
    ggplot(aes(x = weekday, y = avg_sleep)) +
    geom_col(fill = "#529EEF", color = "#529EEA") +
    geom_text(aes(label = ifelse(avg_sleep == max(avg_sleep),
                                 paste("max =", round(avg_sleep,2), "hours"),
                                 ifelse(avg_sleep == min(avg_sleep),
                                        paste("min =", round(avg_sleep,2), "hours"), ""))),
              vjust = -0.5, color = "black", size = 3) +
    labs(title = "Hours of sleep per Weekday",
         x = "weekday",
         y = "average hours sleep") +
    theme_minimal()
```




The data reveals a consistent sleep pattern during the workweek, with individuals getting the most sleep on Sunday and the least on Tuesday.

Recommendations:

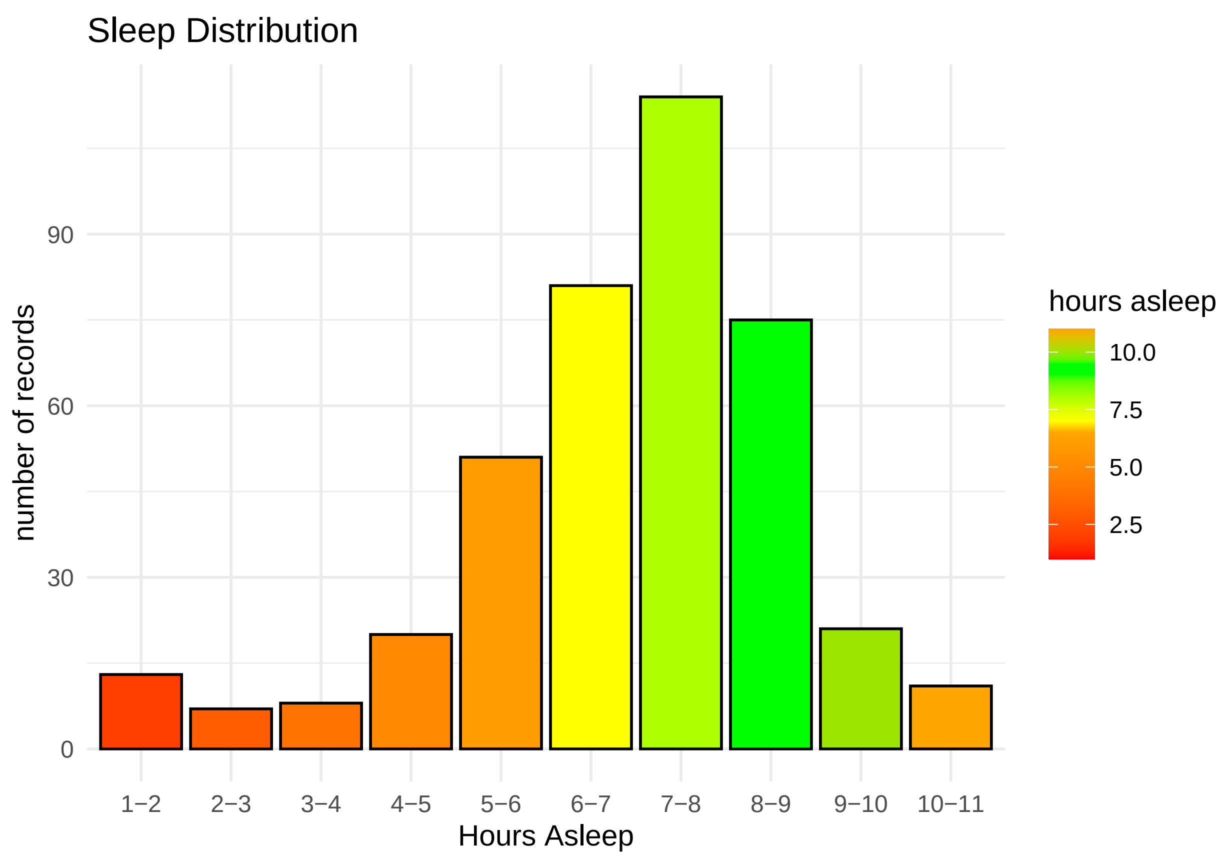
- Aim to even out the rest periods by providing tips on maintaining a stable sleep schedule throughout the week.
- Offer advice on relaxation techniques that could help increase restful sleep on the shorter nights.

#### Sleep distribution

```r
daily_activity_sleep <- daily_activity_sleep %>%
  filter(!is.na(totalminutesasleep)) %>%
```

```
  mutate(hours_asleep_cluster = cut(totalminutesasleep / 60, breaks = seq(0, 14, by = 1), labels = c("0-
```

```
daily_activity_sleep %>%
    filter(!is.na(totalminutesasleep)) %>%
    filter(totalminutesasleep > 60 & totalminutesasleep < 660) %>%
    mutate(hours_asleep_cluster = cut(totalminutesasleep / 60,
                                      breaks = seq(0, 11, by = 1),
                                      labels = c("0-1", "1-2", "2-3", "3-4", "4-5", "5-6", "6-7", "7-8"
                                      include.lowest = TRUE)) %>%
    ggplot(aes(x = hours_asleep_cluster, fill = as.numeric(hours_asleep_cluster))) +
    geom_bar(color = "black") +
    scale_fill_gradientn(colors = c("red", "orange", "yellow", "green", "green", "orange"),
                         values = c(0, 0.55, 0.6, 0.8, 0.85, 1),
                         limits = c(1, 11)) +
    labs(title = "Sleep Distribution",
         x = "Hours Asleep",
         y = "number of records",
         fill = "hours asleep") +
    theme_minimal()
```




The analysis highlights that a notable portion of sleep records (182 in total) falls within or below the 6-7 hour range, indicating insufficient sleep duration.

Recommendations:

- Consider strategies that reinforce the health benefits of 7-8 hours of sleep, reinforcing this as the standard.
- Investigate the causes of both very short and very long sleep durations to offer targeted guidance for

those outliers.

```r
daily_activity_sleep %>%
  filter(!is.na(totalminutesasleep)) %>%
  mutate(hours_asleep_cluster = cut(totalminutesasleep / 60, breaks = seq(0, 10, by = 1), labels = c("0-
  group_by(hours_asleep_cluster) %>%
  summarise(num_records = n()) %>%
  filter(as.character(hours_asleep_cluster) <= "6-7") %>%
  summarise(total_records_6_to_7_hours_or_lower = sum(num_records))
```

```
## # A tibble: 1 x 1
##   total_records_6_to_7_hours_or_lower
##                                 <int>
## 1                                 182
```

### Time in Bed Without Sleep

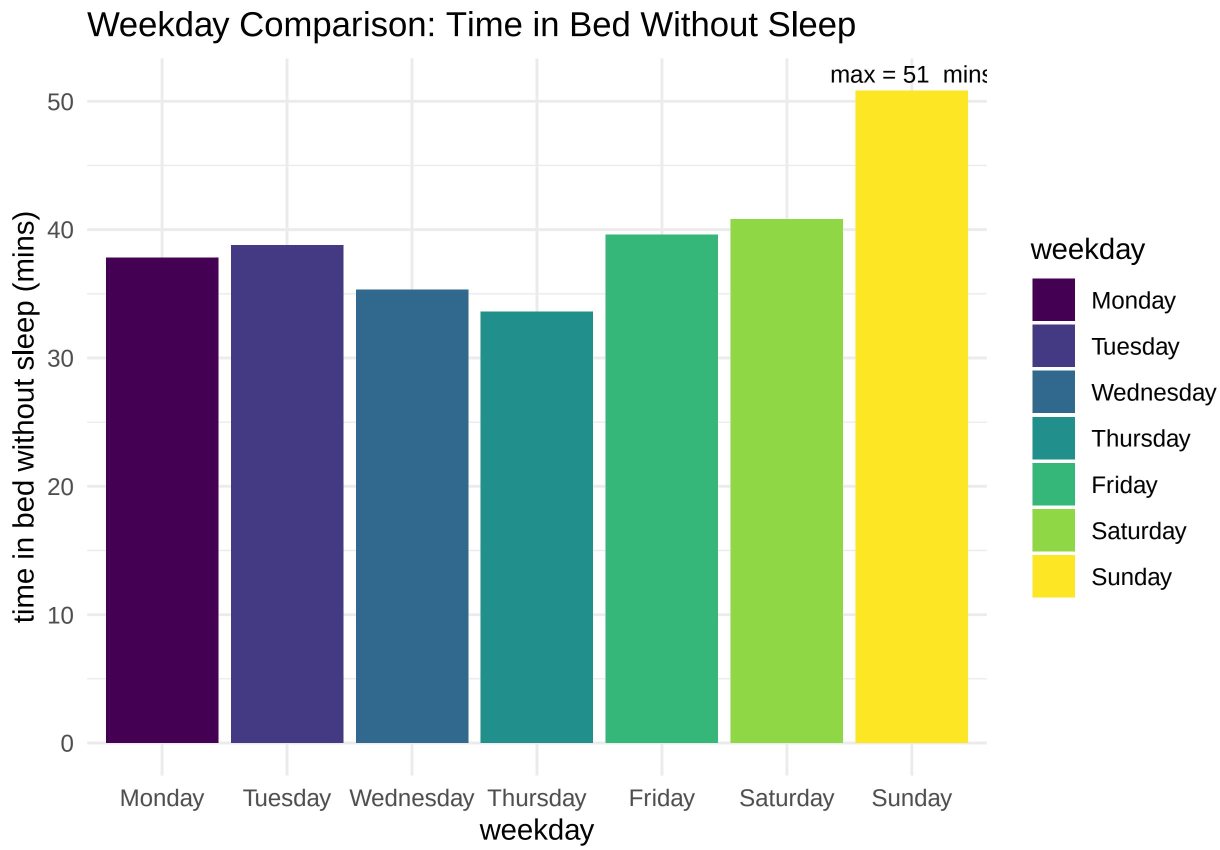
```r
daily_activity_sleep$bedtimegap <- daily_activity_sleep$totaltimeinbed - daily_activity_sleep$totalminu
```

```r
avg_bedtime_gap <- daily_activity_sleep %>%
    na.omit() %>%
    group_by(weekday) %>%
    summarise(avgbedtimegap = mean(bedtimegap)) %>%
  mutate(avgbedtimegap = round(avgbedtimegap,1))
```

Creating a factor

```r
avg_bedtime_gap$weekday <- factor(avg_bedtime_gap$weekday, levels = c("Monday", "Tuesday", "Wednesday",
```

```r
ggplot(avg_bedtime_gap, aes(x = weekday, y = avgbedtimegap, fill = weekday)) +
    geom_col() +
    geom_text(aes(label = ifelse(avgbedtimegap == max(avgbedtimegap),
                                 paste("max =",round(avgbedtimegap), " mins"), "")),
              vjust = -0.5, color = "black", size = 3) +
    labs(
        title = "Weekday Comparison: Time in Bed Without Sleep",
        x = "weekday",
        y = "time in bed without sleep (mins)"
    ) +
    theme_minimal()
```




The analysis reveals that Sunday has the highest average time in bed without sleep, reaching a maximum of 51 minutes, suggesting an opportunity to promote quality time in bed through initiatives such as reducing blue screen exposure before bedtime and encouraging activities like reading or relaxation.

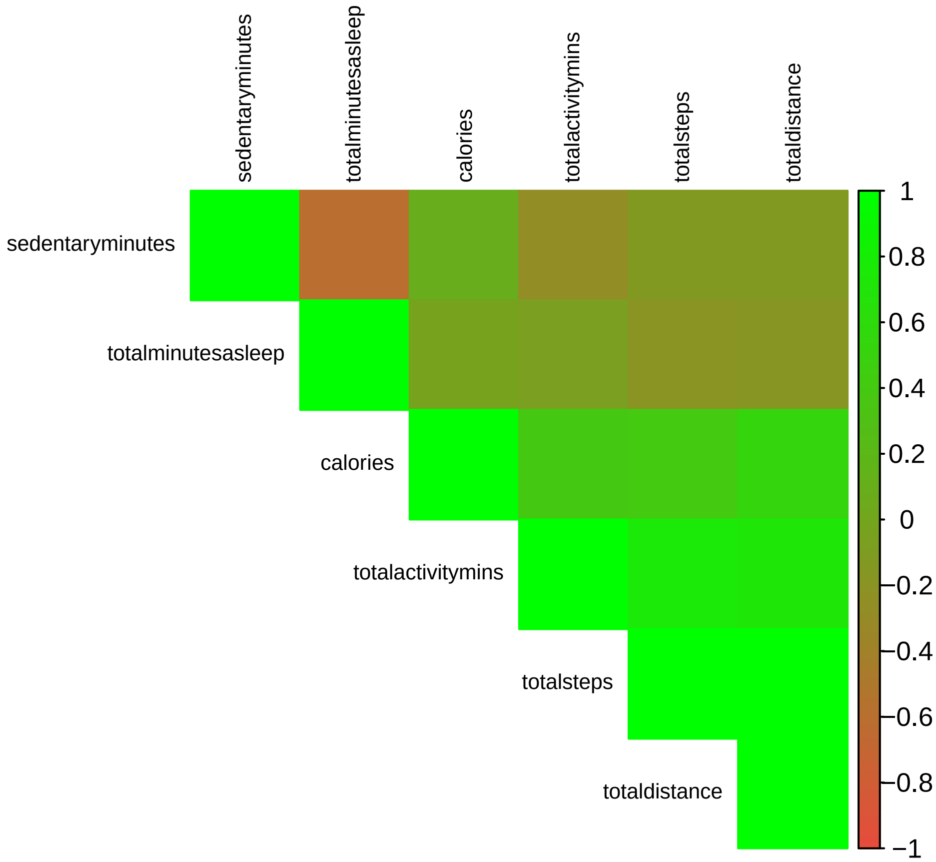
### Correlation matrix

```
daily_activity_sleep$totalactivitymins = daily_activity_sleep$veryactiveminutes + daily_activity_sleep$

correlation_matrix <- daily_activity_sleep %>%
  select(totalsteps, totaldistance, totalminutesasleep, totalactivitymins, calories, sedentaryminutes) %
  cor()
```

```
color_palette <- colorRampPalette(c("#e74c3c", "green"))(100)

corrplot(correlation_matrix,
         method = "color",
         type = "upper",
         order = "hclust",
         tl.cex = 0.7,
         tl.col = "black",
         col = color_palette)
```




The chart shows that sedentary minutes are negatively correlated with total minutes asleep, suggesting that more sleep is associated with fewer sedentary minutes. Additionally, there is a positive correlation between time asleep and calories burned.

Recommendations:

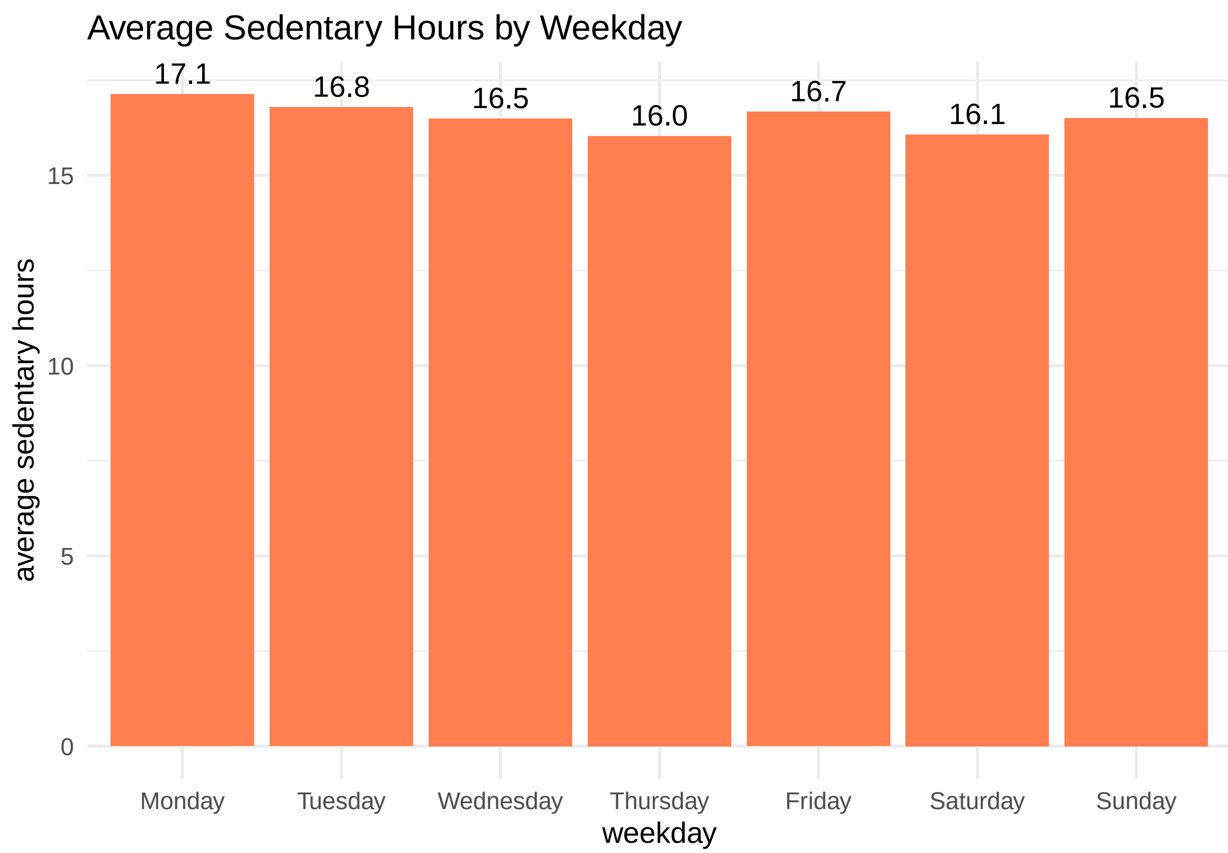
- Develop features that leverage correlations to provide personalized health insights and recommendations.
- Use insights from sedentary minutes and sleep data to encourage less sedentary behavior and more restful nights.

### Average sedentery time

```
daily_activity %>%
  na.omit() %>%
  group_by(weekday) %>%
  summarise(avg_sed = mean(sedentaryminutes) / 60) %>%
  ggplot(aes(x = weekday, y = avg_sed, fill = "coral")) +
  geom_col() +
  geom_text(aes(label = sprintf("%.1f", avg_sed)), vjust = -0.5) +  # Add labels with average values
  scale_fill_manual(values = "coral", guide = FALSE) +  # Remove the legend
  labs(title = "Average Sedentary Hours by Weekday",
       x = "weekday",
       y = "average sedentary hours") +
  theme_minimal()
```

```
## Warning: The `guide` argument in `scale_*()` cannot be `FALSE`. This was deprecated in
## ggplot2 3.3.4.
```

```
## i Please use "none" instead.
## This warning is displayed once every 8 hours.
## Call `lifecycle::last_lifecycle_warnings()` to see where this warning was
## generated.
```




```
daily_activity %>%
  na.omit() %>%
  summarise(average_sedentary_hours = mean(sedentaryminutes) / 60) %>%
  mutate(average_sedentary_hours = round(average_sedentary_hours,2))
```

```
## # A tibble: 1 x 1
##   average_sedentary_hours
##                     <dbl>
## 1                    16.5
```

Based on the data, the average amount of sedentary time during the week appears to be notably high, at 16.5 hours.

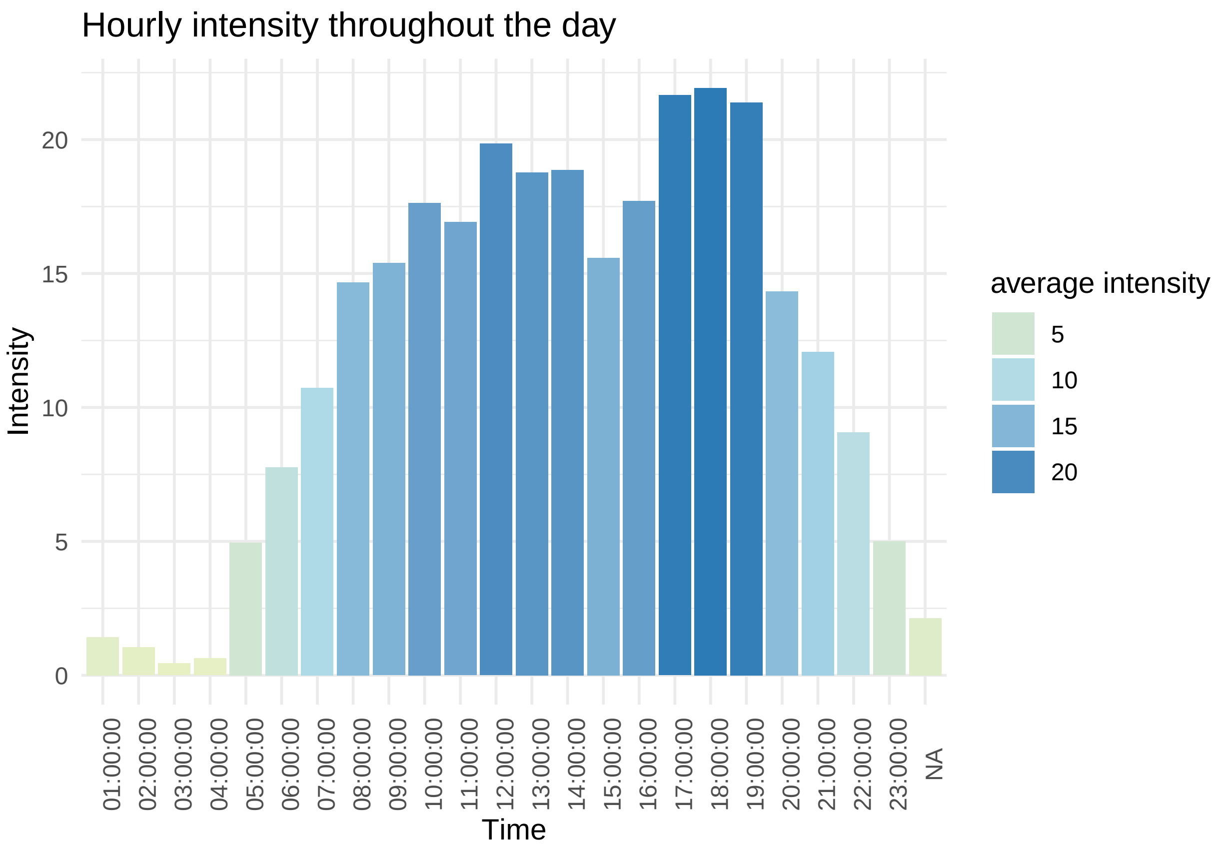
Recommendations: * Introduce short, regular activity breaks during the day to reduce sedentary hours, especially at the beginning of the week. * Encourage weekend active outings to decrease the relatively high sedentary time on these days.

### Hourly intensities throughout the day

```
hourly_intensities <- hourly_intensities %>%
  separate(date_time, into = c("date", "time"), sep= " ")
```

**Average total intensity vs. time**

```
hourly_intensities %>%
  group_by(time) %>%
  summarize(average_intensity = mean(totalintensity)) %>%
  ggplot() +
  geom_col(mapping = aes(x = time, y = average_intensity, fill = average_intensity)) +
  labs(
    title = "Hourly intensity throughout the day",
    x = "Time",
    y = "Intensity",
    fill = "average intensity") +
  scale_fill_gradientn(
    colours = c("#E7F0C3", "#ABD9E9", "#2C7BB6"),  # Modern color palette
    values = c(0, 0.5, 1),  # Custom scale points
    guide = "legend"  # Show legend
  ) +
  theme_minimal() +
  theme(axis.text.x = element_text(angle = 90))
```




The data shows a notable rise in activity intensity during the midday hours, peaking between 12 PM and 2 PM and then 5 PM to 7 PM, which then tapers off as the evening progresses.

Recommendations:

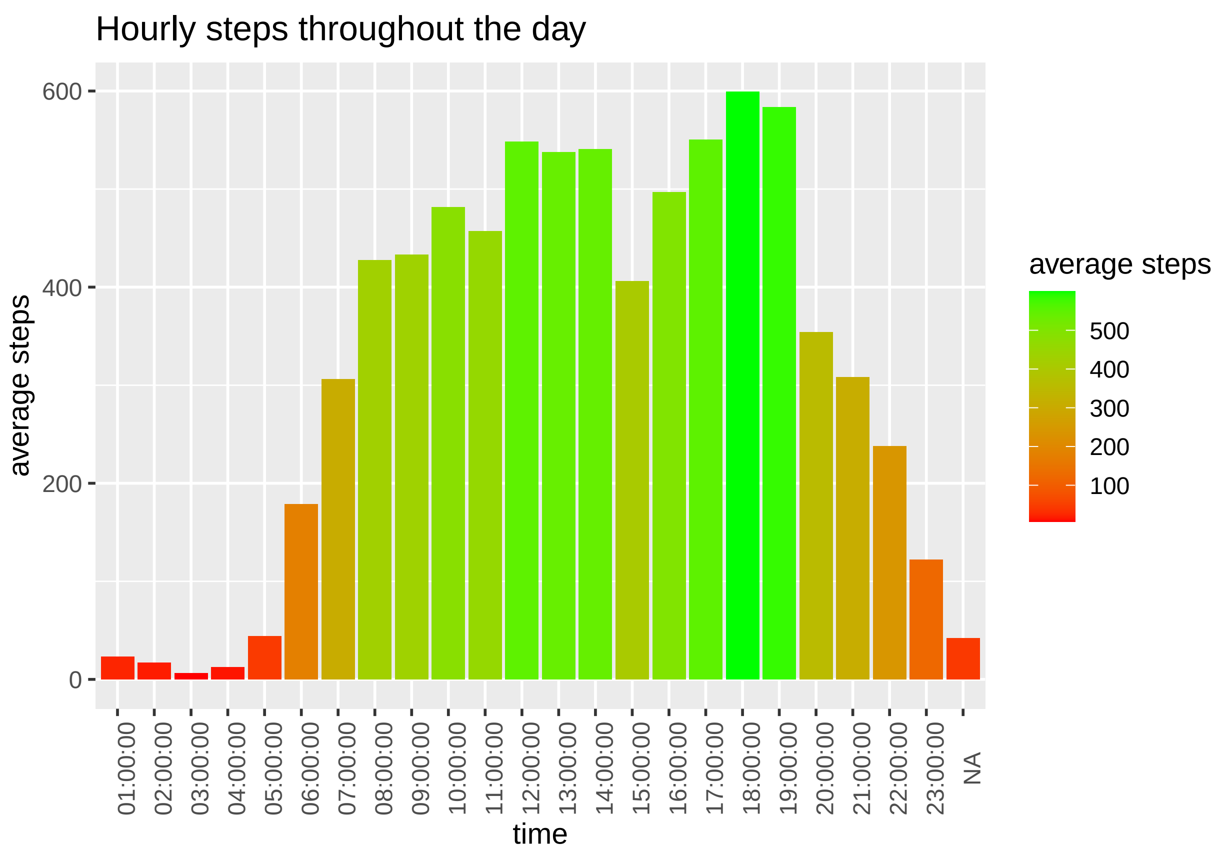
- Explore opportunities for morning and evening activity initiatives to extend the peak intensity beyond the afternoon.
- Promote relaxation and recovery strategies post-peak to facilitate a healthy balance between activity

and rest.

**Hourly steps**

```
hourly_steps <- hourly_steps %>%
  separate(date_time, into = c("date","time"), sep = " ") %>%
  mutate(date = ymd(date))

hourly_steps %>%
  group_by(time) %>%
  summarize(average_steps = mean(steptotal)) %>%
  ggplot() +
  geom_col(mapping = aes(x=time, y = average_steps, fill = average_steps)) +
  labs(
    title = "Hourly steps throughout the day",
    x="time",
    y="average steps",
    fill = "average steps") +
  scale_fill_gradient(low = "red", high = "green")+
  theme(axis.text.x = element_text(angle = 90))
```

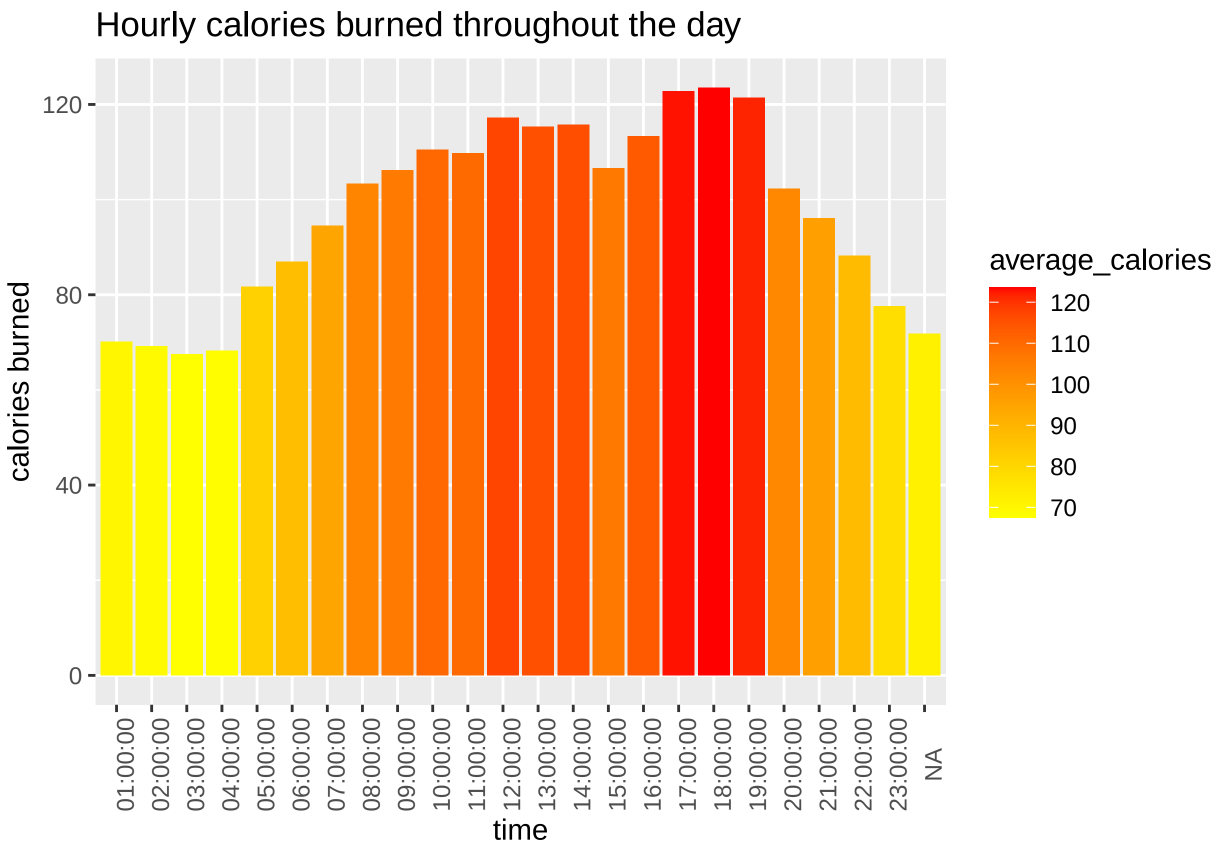



There's a clear surge in step count from mid-morning to late afternoon, mirroring the intensity pattern observed earlier, with peak activity around 1 PM to 4 PM.

**Hourly calories**

```r
hourly_calories <- hourly_calories %>%
    separate(date_time, into = c("date", "time"), sep = " ")

hourly_calories %>%
  group_by(time) %>%
  summarize(average_calories = mean(calories)) %>%
  ggplot() +
  geom_col(mapping = aes(x=time, y = average_calories, fill = average_calories)) +
  labs(title = "Hourly calories burned throughout the day", x="time", y="calories burned") +
  scale_fill_gradient(low = "yellow", high = "red")+
  theme(axis.text.x = element_text(angle = 90))
```




Calorie burn data spikes from late morning to early evening, displaying a robust correlation with the previously noted trends in activity intensity and step count.

Recommendation:

- Suggest engaging in light, calorie-burning activities for the early morning hours to kick-start the day's metabolism.

### User types

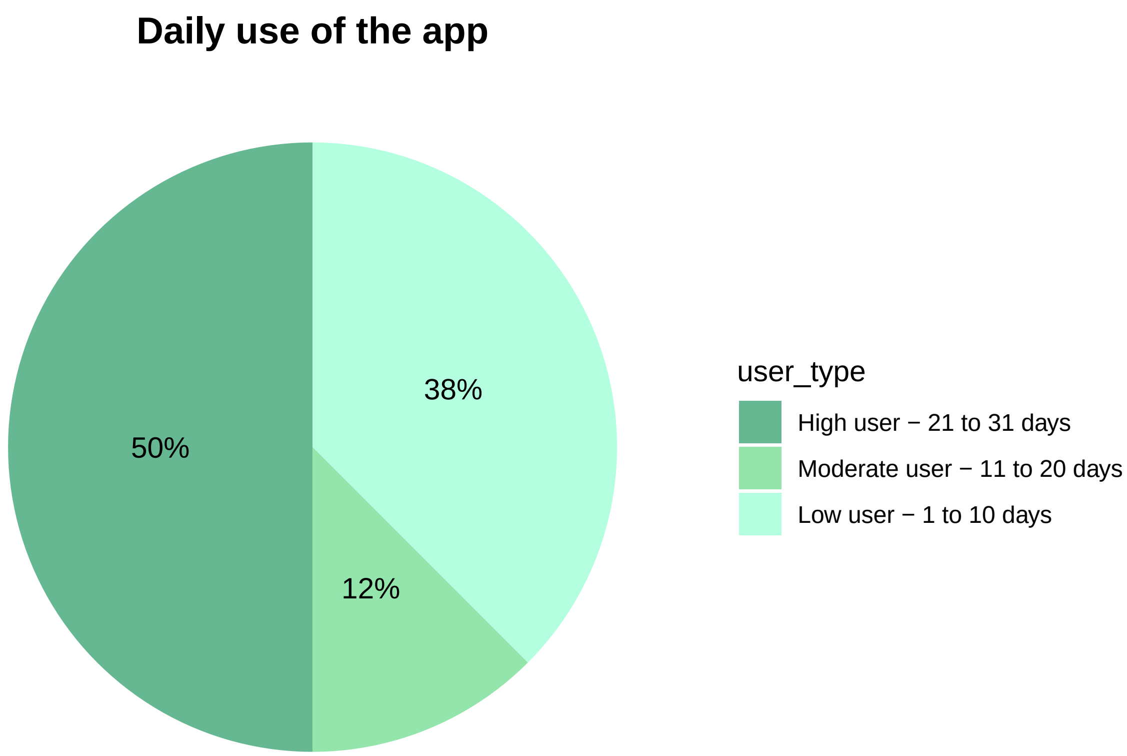
```r
daily_use <- daily_activity_sleep %>%
  group_by(id) %>%
  summarize(days_used=sum(n())) %>%
  mutate(user_type= case_when(
    days_used >= 1 & days_used <= 10 ~ "low user",
    days_used >= 11 & days_used <= 20 ~ "moderate user",
```

```
    days_used >= 21 & days_used <= 31 ~ "heavy user",
  ))

daily_use_percent <- daily_use %>%
  group_by(user_type) %>%
  summarise(total = n()) %>%
  mutate(totals = sum(total)) %>%
  group_by(user_type) %>%
  summarise(total_percent = total / totals) %>%
  mutate(labels = scales::percent(total_percent))

daily_use_percent$user_type <- factor(daily_use_percent$user_type, levels = c("heavy user", "moderate us

daily_use_percent %>%
  ggplot(aes(x = "",y = total_percent, fill = user_type)) +
  geom_bar(stat = "identity", width = 1)+
  coord_polar("y", start=0)+
  theme_minimal()+
  theme(axis.title.x= element_blank(),
        axis.title.y = element_blank(),
        panel.border = element_blank(),
        panel.grid = element_blank(),
        axis.ticks = element_blank(),
        axis.text.x = element_blank(),
        plot.title = element_text(hjust = 0.5, size=14, face = "bold")) +
  geom_text(aes(label = labels),
            position = position_stack(vjust = 0.5))+
  scale_fill_manual(values = c( "#65B891","#93E5AB","#B5FFE1"),
                    labels = c("High user - 21 to 31 days",
                               "Moderate user - 11 to 20 days",
                               "Low user - 1 to 10 days"))+
  labs(title="Daily use of the app")
```




The pie chart vividly segments smart device usage into three categories, showing half of the users as high users, which suggests a strong engagement with the device.

Recommendations:

- Design a rewards program to maintan loyalty of high users.
- Develop targeted programs to convert moderate and low users into more consistent users.